Secure Shell – SSH, zabezpečené připojení

Pozor, útok! (6. díl)

Stále rostoucí snaha a vzrůstající počet úspěšných pokusů hackerů (snaží se o prolomení ochrany s cílem dokázat sobě a svému odbornému okolí svoje znalosti a dovednosti) či crackerů (snaží se prolomit ochranu s cílem nějakým způsobem vědomě poškodit napadený subjekt) o prolomení bezpečnostních vlastností jednotlivých způsobů ochrany privátních dat vyskytujících se na internetu vedou přední návrháře protokolů a standardů k zamyšlení, jak maximálně zvýšit bezpečnost těchto dat a přitom nevytvořit takové bezpečnostní protokoly, jejichž prolomení by sice bylo problematické a takřka nemožné, ale současně by s touto velkou výhodou také vzrostla doba nutná k obsluze jednotlivých bezpečnostních mechanismů užívaných těmito novými řešeními. Jedním z protokolů, které se snaží najít optimální hranici mezi těmito dvěma základními požadavky na zabezpečení přenosu dat, je protokol SSH.

Úvod do SSH

Stejně jako v předchozích dílech seriálu se i tentokrát nejprve seznámíme se základními informacemi a historickými souvislostmi z vývoje protokolu SSH.

Společnost SSH Communications Security vyvinula Secure Shell a SSH protokol – tedy technologie, které se v současné době stávají standardem pro šifrování tzv. “terminal connections” přes internet. Tyto technologie jsou široce rozšířeny především mezi síťovými administrátory, jimž poskytují tři základní utility: slogin, ssh a scp.

Secure Shell je bezpečný přihlašovací program, který změnil vzdálené řízení síťových hostitelů přes internet. Nahrazuje tedy důvěrně známé programy, jako jsou *telnet, rlogin* a *ftp*.

Jedním z důvodů, které vedly k nahrazení těchto programů, byla snaha odstranit velké bezpečnostní riziko, jemuž je uživatel při používání těchto programů nevědomky vystaven. Zmíněné programy totiž přenášejí uživatelská jména nebo hesla přes síť ve formě známé jako “clear text”. Užití tohoto způsobu přenosu dat se logicky stalo pro útočníky poměrně jednoduchým způsobem, jak získat uživatelská jména spolu s odpovídajícími hesly. Tím se dostávala útočníkům do rukou zbraň, kterou neváhali zneužít při svých nekalých aktivitách.

Vývojářům tohoto nového protokolu (SSH) se podařilo vytvořit produkt, který reagoval na negativní vlastnosti výše popsaných programů a získal charakteristické znaky, kterými jsou: automatická autentizace uživatelů – to znamená, že již žádná hesla nejsou přenášena ve zmiňované “clear text” formě vícečetné autentizační metody – reakce na útoky známé jako “spoofing identity”;autentizace také na konci spojení – autentizace serveru a klientu pro zlepšení ochrany například proti “Trojskému koni” apod.šifrování a komprese dat – sloužící k zajištění vyšší bezpečnosti a rychlosti přenosu;bezpečný přenos souborů – užitím tunelování a šifrovaní libovolného spojení.

Architektura SSH

Jak již víme, SSH je protokol pro zabezpečené vzdálené přihlašování (remote login) a pro užití ostatních bezpečných síťových služeb přes jinak nezabezpečené sítě.

Tento protokol se skládá ze tří hlavních stavebních kamenů, kterými jsou následující protokoly (viz obr. 1):

Transport layer protocol (SSH-TRANS) – tento protokol může být užíván jako základ pro řadu bezpečných síťových služeb. Poskytuje serveru autentizaci, utajení a integritu. Pomocí tohoto protokolu jsou dohodnuty algoritmy veřejných klíčů, metody výměny klíčů, symetrické šifrovací algoritmy, algoritmy ověřující zprávy a hašovací algoritmy. Volitelně může tento protokol poskytnout i kompresi.

User authentication protocol (SSH-USERAUTH) – slouží pro potřeby autentizace klientu serverem (host-based client authentication). Tato autentizace může probíhat ve dvou cestách. První z nich je tzv. **Password authentication**. V tomto modu se SSH chová téměř identicky jako *telnet*. V průběhu vytváření nového spojení je uživatel dotázán na heslo, podle něhož je dále buď přihlášen, či zamítnut systémem. Heslo je ovšem v tomto případě nejprve zašifrováno před odesláním přes síť a následně dešifrováno vzdáleným hostitelem. Druhou variantou je tzv. **RSA**

Authentication. V tomto modu je vytvořen pár veřejných a tajných klíčů. Po vytvoření veřejného klíče je tento klíč umístněn na vzdálený host, ke kterému se chce klient přihlásit. To je velmi podobné jako užívání *.rhosts* file při vytváření spojení přes *rlogin.* Další možností je užívat tzv. passphrase, spolu s veřejným klíčem.

Connection protocol (SSH-CONN) – rozděluje zašifrovaný tunel do několika logických kanálů. Například v jednom z těchto logických kanálů může poskytovat tzv. interaktivní "sezení/relaci" při přihlášení (interactive login session) a v druhém například vzdálené provádění příkazů.

Průběh autentizace

Nejdříve si zjednodušeně popíšeme, jak probíhá autentizace serveru klientem – tzv. **Server Host Authentication**.

Poté, co klient vyšle požadavek o spojení se serverem, jsou v dalším kroku vzájemně předány informace o protokolech a verzích. Po této výměně se vytvoří nový klíč serveru (veřejný a tajný), který je pravidelně generován a držen v paměti. Tento veřejný klíč (Server public key) je dále zaslán klientu spolu s hostitelským veřejným klíčem (Host public key), spolu s cookie a informacemi o šifrách. Klient v dalším kroku zkontroluje, zda Host public key patří mezi známé klíče. Pokud ne, zeptá se, jestli má pokračovat. Následuje-li kladná odpověď, přidá veřejný klíč do *~/.ssh/ssh_known_hosts.* Je-li pak vše v pořádku (kladná odpověď i při případné změně Host ID), je v dalším kroku vytvořen Session ID. Postup tvorby tohoto ID lze přirovnat k funkci, jejímiž vstupy jsou veřejné klíče (Host and Server public key) a cookie. V dalším kroku je vytvořen klíč relace (Session key). Tento zašifrovaný klíč relace je spolu s kopií cookie zaslán serveru. Server následně vygeneruje tzv. "Cipher key".

Pozn.: Proces tvorby Session ID a Cipher key je navržen tak, aby mohl probíhat na obou stranách spojení. Veškerá ostatní komunikace je pak již šifrována tímto klíčem.

Nyní se podíváme, jakým způsobem probíhá tzv. **RSA Client – Host Authentication**.

Nejprve zašle klient serveru svůj Client Host public key a svůj hostname. Po obdržení těchto informací server zkontroluje, zda je Client hostname buď v */etc/hots.equiv,* nebo v *~/.rhosts*. Dále kontroluje, zda je Client Host public key v *ssh_known_hosts* v */etc,* nebo v *~/.ssh*. Obě kontroly musí dopadnout dobře. V dalším kroku zašifruje náhodné číslo (Random Number – RN) pomocí Client Host public key. Klient v dalším kroku dešifruje RN pomocí tajného klíče (Client Host private key) a vypočítá MD5 kontrolní součet pro (RN + Session ID). Server v dalším kroku zkontroluje, zda se jím vypočítaný MD5 kontrolní součet shoduje s klientským.

Pozn.: Obdobným způsobem jako v předchozím odstavci (SSH-USERAUTH) probíhá tzv. **User/Password Authentication**.

Závěr

Protokol SSH můžeme rovněž s klidným svědomím zařadit mezi protokoly, jejichž pomocí budeme svěřovat svá data do pomyslných všeobjímajících rukou počítačových sítí, a to zejména proto, že jsou využívány dobře známé a prověřené algoritmy pro šifrování, integritu a veřejné klíče. Všechny tyto algoritmy jsou dohodnuty a v případě poškození či prolomení jednoho algoritmu stačí pouze "přepnout" na jiný algoritmus bez potřeby modifikovat základní protokol.

Čtenáři, kteří rádi experimentují v dané problematice, si mohou stáhnout například ze serveru *www.ssh.fi*: SSH® Secure Shell™ 2.1 BETA for Windows, nebo ze stránek společnosti Data Fellows: F-Secure SSH Client (viz obr. 2).

Milan Pinte I pinte@atlas.cz